AIL861/ELL8299/ELL881: Advanced Large Language Models

End-Semester Examination (Semester I, 2025-26), **IIT Delhi**

Total Marks: 50 Time: 2 hours

Section-A (10 marks)

1. **(Write the numerical answer with proper justification)** Assume you are performing the alignment of a language model $\mathcal{M}$ with human feedback using RLHF (Reinforcement Learning with Human Feedback) and the PPO-loss. **How many language models are required for this process, including** $\mathcal{M}$? *(2 marks)*

> **Answer:** 4
> **Justification:** A standard PPO-based RLHF setup typically requires four models:
> (a) **Policy Model (Actor):** The model M being trained.
> (b) **Reference Model:** A frozen copy of M to compute the KL-divergence penalty.
> (c) **Reward Model:** To assign scalar rewards to the outputs.
> (d) **Value Model (Critic):** To estimate the value function for advantage estimation in PPO.
> Sometimes, in practical use-cases, a separate value/critic model is not used. So, **3 is also an acceptable answer**.

2. **(Write True/False with proper justification)** In BERT, the `[SEP]` token is used to capture sequence information and is utilized in classification tasks. *(1 mark)*

> **Answer:** False
> **Justification:** The `[SEP]` token is used primarily as a separator to distinguish between two sentences in the input (e.g., for Next Sentence Prediction). The token utilized for aggregate sequence representation and classification tasks is the `[CLS]` token.

3. **(Write ALL options which are correct)** For pre-training of encoder-decoder models, which statement(s) is/are true? *(1 mark)*
(A) The encoder attends bidirectionally to its whole input.
(B) The decoder conditions on earlier decoder tokens and encoder outputs.
(C) Unlabelled text is turned into a supervised task via a noising scheme.
(D) Training relies on a next-sentence-prediction loss.

> **Answer:** (A), (B), (C)
> **Justification:**
> - (A) is true: Encoders use bidirectional self-attention.
> - (B) is true: Decoders are autoregressive (condition on earlier tokens) and condition on encoder outputs via cross-attention.
> - (C) is true: Objectives like span corruption (T5) turn unlabelled text into supervised-style tasks.
> - (D) is false: Next-sentence-prediction is a BERT (Encoder-only) objective, not standard for Encoder-Decoder models like T5 or BART.

4. **(Write True/False with proper justification)** In Pipeline Parallelism, increasing the number of micro-batches (m) while keeping the number of pipeline stages (p) constant increases the ratio of the *bubble* (idle time) to the total training time. *(1.5 marks)*

> **Answer:** False
> **Justification:** The bubble fraction in pipeline parallelism is approximately given by $\frac{p-1}{m}$, where p is the number of pipeline stages and m is the number of micro-batches. Therefore, increasing m (the denominator) decreases the ratio of the bubble to total training time.

5. **(Write True/False with proper justification)** Flash Attention achieves speedups primarily by reducing the number of Floating Point Operations (FLOPs) required to compute the attention output compared to standard attention. *(1.5 marks)*

> **Answer:** False
> **Justification:** Flash Attention does not reduce FLOPs (in fact, FLOPs may slightly increase due to re-computation in the backward pass). Its speedup comes from being **IO-aware**: it reduces the number of memory accesses (reads/writes) to the slow High Bandwidth Memory (HBM) by tiling computations in the faster SRAM.

6. **(Write the numerical answer with proper justification)** In a decentralized agentic framework with 8 active agents, how many communications are possible in a worst case setup? *(2 marks)*

> **Answer:** 56
> **Justification:** In a worst-case decentralized setup (fully connected mesh topology), every agent can communicate with every other agent. For $N = 8$ agents, the number of directed communication channels is $N \times (N - 1)$.
> $$8 \times 7 = 56$$
> **Marks have also been given to those who have calculated 28 assuming undirected communication.**

7. **(Write ALL options which are correct)** Which of the following statement(s) correctly describe(s) the features of agentic LLM systems? *(1 mark)*
(A) Agentic systems enable LLMs to interact autonomously with external tools and APIs.
(B) Agentic workflows perform tasks strictly in a zero-shot manner without iterative refinement.
(C) Agentic protocols like MCP and A2A provide interoperability standards for agents and tools.
(D) Memory management frameworks like MemGPT allow agents to maintain long-term context through paging.

> **Answer:** (A), (C), (D)
> **Justification:**
> - (A) is true: Agents use tools.
> - (B) is false: Agentic workflows often use iterative refinement (e.g., ReAct, Reflection) rather than strict zero-shot execution.
> - (C) is true: Model Context Protocol (MCP) and Agent-to-Agent (A2A) are interoperability standards.
> - (D) is true: MemGPT uses OS-like paging for long-term context.

Section-B (40 marks)

Question 1: Knowledge Distillation (3 marks)

(a) What is *Meta Knowledge Distillation*? *(1 mark)*

> Meta Knowledge Distillation is a distillation paradigm where the teacher helps guide the student's training strategies, such as data selection or curriculum learning. It also often involves meta-learning distillation hyperparameters (e.g., temperature in the softmax) dynamically during training to optimize the student's performance on a validation set.

(b) Explain with mathematical formalism how *Reverse KL-Divergence* is different from *Forward KL-Divergence*. What is the benefit of using Reverse KL-Divergence for knowledge distillation in LLMs? *(1+1=2 marks)*

> **Mathematical Formalism:** Let P be the teacher distribution and Q be the student distribution.
>
> – **Forward KL-Divergence:** $D_{KL}(P||Q) = \sum_y P(y|x) \log \frac{P(y|x)}{Q(y|x)}$. This objective is *mean-seeking* (or zero-avoiding). It forces the student Q to cover the entire support of the teacher P to avoid infinite cost when $P(y|x) > 0$ and $Q(y|x) \to 0$.
>
> – **Reverse KL-Divergence:** $D_{KL}(Q||P) = \sum_y Q(y|x) \log \frac{Q(y|x)}{P(y|x)}$. This objective is *mode-seeking* (or zero-forcing). The student Q minimizes cost by assigning low probability to regions where the teacher P has low probability.
>
> **Benefit for LLMs:** LLMs have complex output distributions with a vast long tail of low-probability sequences. Forward KLD forces the student to overestimate these low-probability regions (voids) to cover the teacher's full support, leading to low-quality or nonsensical generation. Reverse KLD encourages the student to focus on the teacher's major modes (high-quality outputs) and ignore the long tail, resulting in more precise and correct generations.

Question 2: Retrieval-based Language Models (6 marks)

(a) Briefly explain the difference between *sparse retrieval* and *dense retrieval* methods with examples of both. *(2 marks)*

– **Sparse Retrieval:** Represents queries and documents as high-dimensional sparse vectors based on word occurrences/frequencies. It relies on lexical matching (exact keyword overlap) between the query and documents. *Examples:* TF-IDF, BM25.

– **Dense Retrieval:** Uses neural network encoders (like BERT) to map queries and documents into lower-dimensional continuous dense vectors. Retrieval is performed using Maximum Inner Product Search (MIPS) to find semantic similarity, even if there is no exact keyword overlap. *Examples:* DPR (Dense Passage Retrieval), Contriever.

(b) Assume we have a set of retrieval-augmented generation (RAG) models which use the input sequence x to retrieve text documents z and use them as additional context when generating the target sequence y. These models have two components: a retriever $p_r(z|x)$, and a generator $p_g(y_i|x, z, y_{1:i-1})$. Assuming top-K approximation for document retrieval and $|y| = N$, write the formal mathematical expressions for $p(y|x)$ for (i) **RAG-Sequence**, and (ii) **RAG-Token** models. *(2+2=4 marks)*

(i) RAG-Sequence: The model uses the *same* retrieved document z to generate the complete sequence. The document is treated as a single latent variable that is marginalized to get the sequence probability.

$$p(y|x) \approx \sum_{z \in \text{top-K}(p_r(\cdot|x))} p_r(z|x) \prod_{i=1}^{N} p_g(y_i|x, z, y_{1:i-1})$$

(ii) RAG-Token: The model can draw a *different* latent document z for each target token. The marginalization over the top-K documents happens at every token generation step.

$$p(y|x) \approx \prod_{i=1}^{N} \sum_{z \in \text{top-K}(p_r(\cdot|x))} p_r(z|x) p_g(y_i|x, z, y_{1:i-1})$$

Question 3: LLMs and Tools (6 marks)

(a) Briefly explain the **three steps** involved in *Toolformer* to convert a standard language modeling dataset to a dataset with API calls. *(1.5 marks)*

The three steps to convert a dataset in Toolformer are:

1. **Sample API Calls:** Use the LM with few-shot prompts to generate potential API calls at different positions in the text.
2. **Execute API Calls:** Execute the sampled calls to obtain their return values (results).
3. **Filter API Calls:** Calculate the weighted cross-entropy loss. Keep only the API calls where providing the input and the result reduces the perplexity (loss) of predicting future tokens compared to not doing the call (or doing it without the result).

(b) Considering a language model $\mathcal{M}$, an input x, a function `stop(.)` to check if stop condition is reached, and the prompts p_{gen}, p_{fb} and p_{refine} for initial generation, feedback and refinement, respectively, **write the algorithm for the Self-Refine approach.** *(2 marks)*

Self-Refine Algorithm:

```
1. Initial Generation: $y_0 \leftarrow \mathcal{M}(P_{gen}, x)$
2. Loop $t = 0, 1, \ldots$ do
3.     Feedback: $fb_t \leftarrow \mathcal{M}(P_{fb}, x, y_t)$
4.     Stop Condition: If stop($fb_t$) is true, return $y_t$
5.     Refinement: $y_{t+1} \leftarrow \mathcal{M}(p_{refine}, x, y_0, fb_0, \ldots, y_t, fb_t)$
6. End Loop
7. Return $y_t$
```

(c) Which two transport methods are supported by Model Context Protocol (MCP)? What are the core primitives that a server can expose in MCP? *(1+1.5=2.5 marks)*

- **Transport Methods:** (1) **STDIO** (Standard Input/Output) for local processes, and (2) **HTTP+SSE** (Server-Sent Events) for remote connections.
- **Core Primitives:**
  1. **Tools:** Executable functions the model can invoke (model-controlled).
  2. **Resources:** Read-only data sources like files or database records (application-controlled).
  3. **Prompts:** Reusable templates to structure interactions (user-controlled).

Question 4: LLM Reasoning (7 marks)

(a) Formalize the concept of RL with Verifiable Rewards (RLVR), and explain (with equation(s) of training objective(s)) how it can be used to train specialized reasoning models. *(3 marks)*

RL with Verifiable Rewards (RLVR) is a training framework where a language model (policy π_θ) is optimized using reinforcement learning (e.g., PPO) with a binary reward signal derived from a deterministic verification process. Unlike RLHF, which relies on a learned reward model approximating human preference, RLVR uses ground-truth correctness (e.g., checking if a generated math answer matches the correct solution or if code passes unit tests) as the reward $r(y, y^*)$.
Training Objective: The model maximizes the expected reward of the generated reasoning chains y given input x:

$$J(\theta) = \mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[r(y, y^*)]$$

where $r(y, y^*) = 1$ if the answer is correct, and 0 otherwise. This encourages the model to explore and reinforce reasoning paths that lead to the correct solution.

(b) Monte Carlo Tree Search (MCTS) is used to explore a diverse solution space and create a synthetic dataset to teach structured reasoning to language models. MCTS involves the stages of *selection*, *expansion* and *backpropagation*. In the backpropagation step, the quality score $Q(S_t, a_i)$ of taking action a_i from state S_t is updated recursively based on the reward scores obtained during expansion from the leaf node to the root node. Given the action index $i \in [1, ..., K]$, the rollout-based reward score for the state S_{t+1} to be $R(S_{t+1})$, and the visit count of taking action a_i from S_{t+1} to be $N(S_{t+1}, a_i)$, **write the update equation for $Q(S_t, a_i)$ during backpropagation.** *(2 marks)*

During the backpropagation stage of MCTS, the reward scores obtained during expansion from the leaf node to the root node are backpropagated to recursively update their Q-values. The Q-values of each (state, action) pair is updated using the Q-values and visit counts of the children nodes of $S_{t+1} = (S_t, a)$, along with the rollout-based reward score $R(S_{t+1})$:

$$Q(S_t, a) = \frac{\sum_{i=1}^{K} Q(S_{t+1}, a_i) \cdot N(S_{t+1}, a_i) + R(S_{t+1})}{\sum_{i=1}^{K} N(S_{t+1}, a_i) + 1}$$

(c) Briefly describe the main difference between the parallel and sequential scaling approaches for test-time scaling in LLMs. *(1 mark)*

- **Parallel Scaling** involves generating multiple independent solutions simultaneously (e.g., Best-of-N, Majority Voting) and selecting the best one using a verifier. It is highly parallelizable but limited by the single-shot capability of the model.
- **Sequential Scaling** involves iterative exploration of the solution space (e.g., Tree Search, Chain-of-Thought refinement), where subsequent steps depend on previous states. It allows for error correction and deeper reasoning but incurs higher latency.

(d) Assuming an ideal verifier that always recognizes the correct answer if it appears, and p to be the pass@1 rate, i.e., the probability that a single generation is correct, **what is the probability of success of the Best-of-N strategy in test-time scaling?** *(1 mark)*

The probability of success (at least one correct generation) is:

$$P(\text{success}) = 1 - (1 - p)^N$$

Question 5: Multimodal Models (5 marks)

(a) The standard Transformer receives as input a 1D sequence of token embeddings. However, when dealing with images, a challenge is posed – each image is a matrix of dimension $H \times W \times C$, where H, W and C are respectively the height, width and number of channels of an image.

How does a Vision Transformer (ViT) handle images as input? Specifically state the input format (with dimensions) which goes into the ViT, and whether any architectural changes need to be made to ViTs compared to standard Transformer Encoders. *(2 marks)*

An image $x \in \mathrm{R}^{H \times W \times C}$ is reshaped into a sequence $\mathbf{x_p}$ of flattened 2D patches, such that, $\mathbf{x_p} \in \mathrm{R}^{N \times P^2 C}$, where, $N = \frac{HW}{P^2}$, and (P, P) is the resolution of each image patch. P is a hyperparameter.

There is no change in the architecture of the Transformer Encoder. Each flattened patch is a vector of dimension $(1, P^2C)$, which are linearly embedded (just like there is an embedding for each token) and positional embeddings are added to them. The Transformer uses a vector of size D through all of its layers, with D being the embedding dimension.

(b) Briefly explain the following related to the Bootstrapping Language-Image Pre-training (BLIP) approach for vision-language pre-training:
(i) **Model architecture, specifically the multimodal mixture of encoder-decoder**, and
(ii) **Different components of the pre-training objective**. *(1.5+1.5=3 marks)*

(i) Model Architecture: BLIP uses a **Multimodal Mixture of Encoder-Decoder (MED)** architecture. This unified model can operate in three functionalities:

- **Unimodal Encoder:** Encodes image and text separately. The text encoder is BERT-like with a `[CLS]` token.
- **Image-Grounded Text Encoder:** Injects visual information into the text encoder by inserting a cross-attention layer between the self-attention layer and the feed-forward network for each transformer block.
- **Image-Grounded Text Decoder:** Replaces the bi-directional self-attention layers with causal self-attention layers to generate text given an image.

The text encoder and decoder share all parameters except for the self-attention layers.

(ii) Pre-training Objectives: BLIP is jointly pre-trained with three objectives:

1. **Image-Text Contrastive Loss:** Aligns the feature space of the visual and text transformers by encouraging positive pairs to have similar representations.
2. **Image-Text Matching Loss:** A binary classification task where the image-grounded text encoder predicts whether an image-text pair matches (positive) or not (negative).
3. **Language Modeling Loss:** Activates the image-grounded text decoder to generate textual descriptions given an image by maximizing the likelihood of the text in an auto-regressive manner.

Question 6: Alternative Architectures (6 marks)

(a) Given the linear recurrence: $x_t = \bar{A}\, x_{t-1} + \bar{B}\, u_t$; $y_t = C\, x_t$, write the output y_t in the form of a causal convolution. *(2 marks)*

We start from

$$x_t = \bar{A}x_{t-1} + \bar{B}u_t, \qquad y_t = Cx_t.$$

Unroll:

$$\begin{aligned} x_t &= \bar{A}x_{t-1} + \bar{B}u_t \\ &= \bar{A}(\bar{A}x_{t-2} + \bar{B}u_{t-1}) + \bar{B}u_t \\ &= \bar{A}^2x_{t-2} + \bar{A}\bar{B}u_{t-1} + \bar{B}u_t \\ &\vdots \\ &= \bar{A}^k x_{t-k} + \sum_{j=0}^{k-1} \bar{A}^j \bar{B}\, u_{t-j}. \end{aligned}$$

Take $k = t$:

$$x_t = \bar{A}^t x_0 + \sum_{j=0}^{t-1} \bar{A}^j \bar{B}\, u_{t-j}.$$

Output:

$$y_t = C\bar{A}^t x_0 + \sum_{j=0}^{t-1} C\bar{A}^j \bar{B}\, u_{t-j}.$$

Convolution kernel:

$$K_j = C\bar{A}^j\bar{B}, \qquad j \geq 0.$$

Thus,

$$y_t = \sum_{j=0}^{t-1} K_j\, u_{t-j} \quad (\text{if } x_0 = 0).$$

and explicitly,

$$K = \big[\, C\bar{B},\ C\bar{A}\bar{B},\ C\bar{A}^2\bar{B},\ C\bar{A}^3\bar{B},\ \ldots \,\big].$$

(b) (i) Write down the structure of the HiPPO–LegS matrix A_{HiPPO} and the input vector B_{HiPPO}.
(ii) Why is A_{HiPPO} upper triangular and what does this imply for the memory dynamics?
(iii) Why are the diagonal entries of A_{HiPPO} negative and what does this ensure?
(iv) What is the role of the input vector B_{HiPPO}? *(1+1+1+1=4 marks)*

(i) The HiPPO–LegS matrices are:

$$(A_{\text{HiPPO}})_{kn} = \begin{cases} -\sqrt{(2k+1)(2n+1)}, & n < k, \\ -(k+1), & n = k, \\ 0, & n > k, \end{cases} \qquad B_{\text{HiPPO},k} = \sqrt{2k+1}.$$

(ii) Upper-triangular A_{HiPPO}: slow modes feed fast ones (context $\to$ connectors).
(iii) Negative diagonal: ensures stable decay.
(iv) B_{HiPPO}: injects new token into all memory scales.

Question 7: Physics of LLMs and Interpretability (7 marks)

(a) If you are designing a small-scale language model with a fixed parameter budget specifically to solve long-chain reasoning problems, would you prioritize increasing the **depth** (number of layers) or the **width** (number of heads/hidden dimension) of the model? Justify your choice by explaining the mechanism that limits the performance of a model architecture in this regard. *(1+1=2 marks)*

Choice: Prioritize **depth** of the model (number of layers).
Justification: Depth is the critical factor for solving long-chain reasoning tasks. A transformer layer is limited to a certain amount of sequential computation; therefore, a reasoning chain of length K typically requires a model depth $D \geq K$ to be solved correctly. Increasing width adds memory capacity for facts but does not effectively extend the length of the serial reasoning process the model can execute.

(b) Given an activation $a \in \mathbb{R}^d$ corresponding to a neuron of the language model, explain how a sparse auto-encoder (SAE) operate on it to extract interpretable features. What is the training objective of SAE? *(1.5+1.5=3 marks)*

Given an activation vector $a \in \mathbb{R}^d$, a Sparse Autoencoder (SAE) projects it into a higher-dimensional sparse feature space f using an encoder with a non-linearity (typically ReLU):

$$f(a) = \text{ReLU}(W_{enc}a + b_{enc})$$

It then approximates the original activation $\hat{a}$ using a decoder: $\hat{a}(f) = W_{dec}f + b_{dec}$.
Training Objective: The SAE is trained to minimize a loss function composed of the reconstruction error (usually L2) and a sparsity penalty (L1) on the feature activations f:

$$\mathcal{L} = ||a - \hat{a}||_2^2 + \lambda ||f||_1$$

This encourages the model to represent the dense activation a as a linear combination of a few interpretable feature directions.

(c) Explain the difference between standard SAEs, Transcoders and Crosscoders. *(2 marks)*

- **Standard SAEs:** Trained on activations from a *single* layer to reconstruct the *same* input ($x \to \hat{x}$), isolating features present at that specific point in the network.
- **Transcoders:** Trained to map activations from an input layer to the output of a subsequent sub-layer (e.g., mapping MLP input to MLP output). They effectively replace a specific computation block to interpret the *transformation* of features rather than just their presence.
- **Crosscoders:** Trained on activations from *multiple* layers or models simultaneously. They project these distinct inputs into a *shared* latent feature space to identify shared features and compare differences across layers or model versions.

♣ ♠ ♡ ♢